Juho Hirvonen and Jukka Suomela:

Distributed Algorithms 2020

https://jukkasuomela.fi/da2020/ · September 20, 2023

Chapter 9

Round Elimination

In this chapter we introduce the basic idea of a proof technique, called *round elimination*. Round elimination is based on the following idea. Assume that there exists a distributed algorithm S_0 with complexity T solving a problem Π_0. Then there exists a distributed algorithm S_1 with complexity $T-1$ for solving another problem Π_1. That is, if we can solve problem Π_0 in T communication rounds, then we can solve a related problem Π_1 exactly one round faster—we can "eliminate one round". If this operation is repeated T times, we end up with some algorithm S_T with round complexity 0 for some problem Π_T. If Π_T is not a *trivial* problem, that is, cannot be solved in 0 rounds, we have reached a contradiction: therefore the assumption that Π_0 can be solved in T rounds has to be wrong. This is a very useful approach, as it is much easier to reason about 0-round algorithms than about algorithms in general.

9.1 Bipartite Model and Biregular Trees

When dealing with round elimination, we will consider a model that is a variant of the PN model from Chapter 3. We will restrict our attention to specific families of graphs (see Figure 9.1):

(a) **Bipartite.** The set of nodes V is partitioned into two sets: the *active* nodes V_A and the *passive* nodes V_P. The partitioning forms a proper 2-coloring of the graph, i.e., each edge connects an active

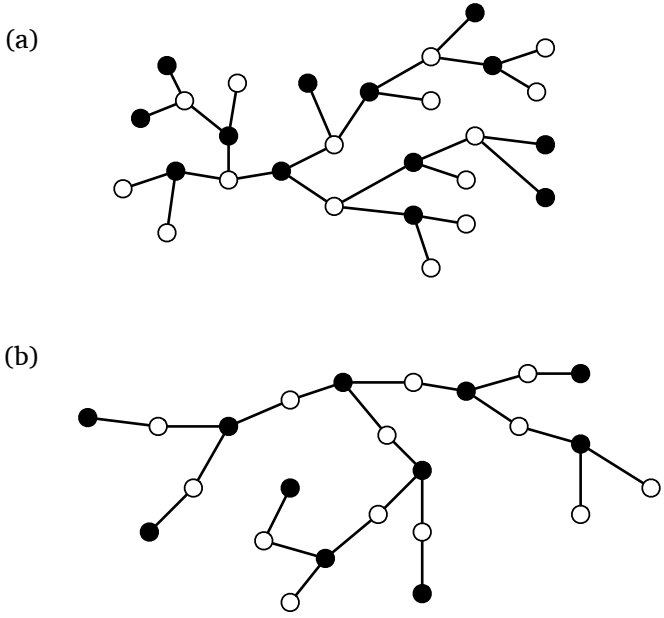

Figure 9.1: The bipartite model; black nodes are active and white nodes are passive. (a) A $(3,3)$-biregular tree. (b) A $(3,2)$-biregular tree.

node with a passive node. The role of a node—active or passive—is part of the local input.

(b) **Biregular trees.** We will assume that the input graphs are *biregular* trees: the graph is connected, there are no cycles, each node in V_A has degree d or 1, and each node in V_P has degree δ or 1. We say that such a tree is (d,δ)-biregular. See Figure 9.1 for an illustration.

9.1.1 Bipartite Locally Verifiable Problem

We consider a specific family of problems, called *bipartite locally verifiable* problems. Such a problem is defined as a 3-tuple $\Pi = (\Sigma, \mathbf{A}, \mathbf{P})$, where:

- Σ is a finite alphabet.
- $\mathbf{A}$ and $\mathbf{P}$ are finite collections of multisets, where each multiset $A \in \mathbf{A}$ and $P \in \mathbf{P}$ consists of a finite number of elements from Σ. These are called the *active* and *passive configurations*.

Recall that multisets are sets that allow elements to be repeated. We use the notation $[x_1, x_2, \ldots, x_k]$ for a multiset that contains k elements; for example, $[1, 1, 2, 2, 2]$ is a multiset with two 1s and three 2s. Note that the order of elements does not matter, for example, $[1, 1, 2] = [1, 2, 1] = [2, 1, 1]$.

In problem Π, each active node $v \in V_A$ must label its incident $\deg(v)$ edges with elements of Σ such that the labels of the incident edges, considered as a multiset, form an element of $\mathbf{A}$. The order of the labels does not matter. The passive nodes do not have outputs. Instead, we require that for each passive node the labels of its incident edges, again considered as a multiset, form an element of $\mathbf{P}$. A labeling $\varphi : E \to \Sigma$ is a solution to Π if and only if the incident edges of all active and passive nodes are labeled according to some configuration.

In this chapter we will only consider labelings such that all nodes of degree 1 accept any configuration: these will not be explicitly mentioned in what follows. Since we only consider problems in (d, δ)-biregular trees, each active configuration will have d elements and each passive configuration δ elements.

9.1.2 Examples

To illustrate the definition of bipartite locally verifiable labelings, we consider some examples (see Figure 9.2).

Edge Coloring. A c-edge coloring is an assignment of labels from $\{1, 2, \ldots, c\}$ to the edges such that no node has two incident edges with the same label.

Consider the problem of 5-edge coloring $(3, 3)$-biregular trees. The alphabet Σ consists of the five edge colors $\{1, 2, 3, 4, 5\}$. The active

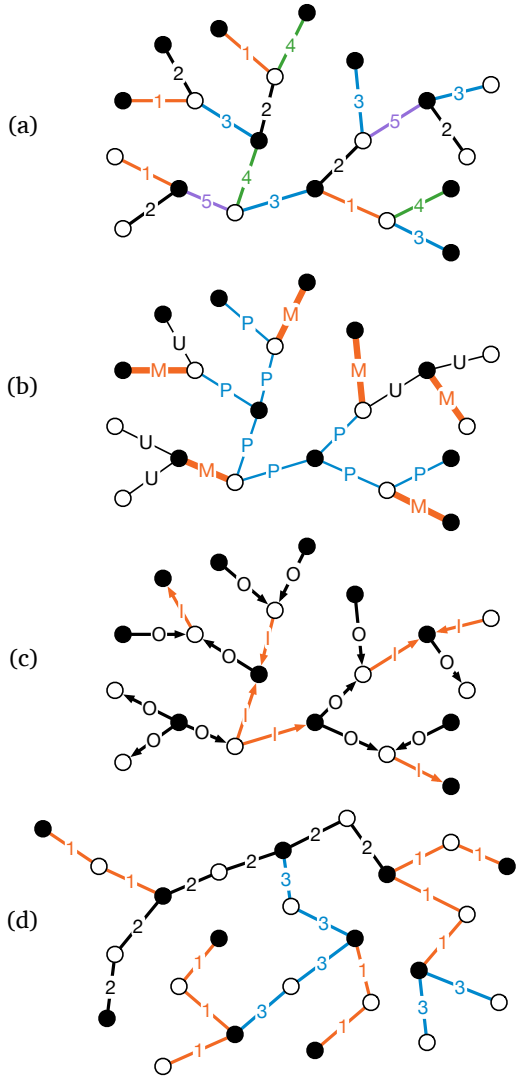

Figure 9.2: Bipartite locally verifiable labeling problems. (a) 5-edge coloring in a $(3,3)$-biregular tree. (b) Maximal matching in a $(3,3)$-biregular tree. (c) Sinkless orientation in a $(3,3)$-biregular tree. (d) Weak 3-labeling in a $(3,2)$-biregular tree.

configurations consist of all multisets of three elements $[x, y, z]$, such that all elements are distinct and come from Σ. The problem is symmetric, and the passive configurations consist of the same multisets:

$$\mathbf{A} = \mathbf{P} = \big\{ [1,2,3], [1,2,4], [1,2,5], [1,3,4], [1,3,5], \\ [1,4,5], [2,3,4], [2,3,5], [2,4,5], [3,4,5] \big\}.$$

Maximal Matching. A maximal matching M is a subset of the edges such that no two incident edges are in M and no edge can be added to M.

Consider maximal matching on $(3,3)$-biregular trees. To encode a matching, we could use just two labels: M for matched and U for unmatched. Such a labeling, however, has no way of guaranteeing maximality. We use a third label P, called a pointer:

$$\Sigma = \{\mathsf{M}, \mathsf{P}, \mathsf{U}\}.$$

The active nodes either output $[\mathsf{M}, \mathsf{U}, \mathsf{U}]$, denoting that the edge marked M is in the matching, or they output $[\mathsf{P}, \mathsf{P}, \mathsf{P}]$, denoting that they are unmatched, and thus all passive neighbors *must* be matched with another active node:

$$\mathbf{A} = \big\{ [\mathsf{M}, \mathsf{U}, \mathsf{U}], [\mathsf{P}, \mathsf{P}, \mathsf{P}] \big\}.$$

Passive nodes must verify that they are matched with at most one node, and that if they have an incident label P, then they also have an incident label M (to ensure maximality). Hence the passive configurations are

$$\mathbf{P} = \big\{ [\mathsf{M}, \mathsf{P}, \mathsf{P}], [\mathsf{M}, \mathsf{P}, \mathsf{U}], [\mathsf{M}, \mathsf{U}, \mathsf{U}], [\mathsf{U}, \mathsf{U}, \mathsf{U}] \big\}.$$

Sinkless Orientation. A *sinkless orientation* is an orientation of the edges such that each node has an edge oriented away from it. That is, no node is a *sink*. We will consider here sinkless orientation in $(3,3)$-biregular trees; leaf nodes can be sinks, but nodes of degree 3 must have at least one outgoing edge.

To encode sinkless orientation, each active node chooses an orientation of its incident edges: outgoing edges are labeled O and incoming

edges I. Thus the alphabet is $\Sigma = \{\mathsf{O}, \mathsf{I}\}$. Each node must have an outgoing edge, so the active configurations are all multisets that contain at least one O:

$$\mathbf{A} = \big\{[\mathsf{O}, x, y] \;\big|\; x, y \in \Sigma\big\}.$$

The passive configurations are similar, but the roles of the labels are reversed: an outgoing edge for an active node is an incoming edge for a passive node. Therefore each passive node requires that at least one of its incident edges is labeled I, and the passive configurations are

$$\mathbf{P} = \big\{[\mathsf{I}, x, y] \;\big|\; x, y \in \Sigma\big\}.$$

Weak Labeling. We will use the following problem as the example in the remainder of this chapter. Consider $(3,2)$-biregular trees. A *weak* 3*-labeling* is an assignment of labels from the set $\{1,2,3\}$ to the edges such that each active node has at least two incident edges labeled with different labels. Each passive node must have its incident edges labeled with the same label. The problem can be formalized as

$$\begin{aligned}
&\Sigma = \{1,2,3\},\\
&\mathbf{A} = \big\{[1,1,2], [1,1,3], [1,2,2], [1,2,3], [1,3,3], [2,2,3], [2,3,3]\big\},\\
&\mathbf{P} = \big\{[1,1], [2,2], [3,3]\big\}.
\end{aligned}$$

9.2 Introducing Round Elimination

Round elimination is based on the following basic idea. Assume that we can solve some bipartite locally verifiable problem Π_0 in T communication rounds on (d,δ)-biregular trees. Then there exists a bipartite locally verifiable problem Π_1, called the *output problem* of Π_0, that can be solved in $T-1$ rounds on (δ,d)-biregular trees. The output problem is uniquely defined, and we refer to the output problem of Π as $\mathrm{re}(\Pi)$. The definition of output problem will be given in Section 9.2.2.

A single round elimination step is formalized in the following lemma.

Lemma 9.1 (Round elimination lemma)**.** *Let* Π *be bipartite locally verifiable problem that can be solved in* T *rounds in* (d, δ)*-biregular trees. Then the output problem* $\mathrm{re}(\Pi)$ *of* Π *can be solved in* $T - 1$ *rounds in* (δ, d)*-biregular trees.*

9.2.1 Impossibility Using Iterated Round Elimination

Lemma 9.1 can be iterated, applying it to the output problem of the previous step. This will yield a sequence of $T + 1$ problems

$$\Pi_0 \to \Pi_1 \to \cdots \to \Pi_T,$$

where $\Pi_{i+1} = \mathrm{re}(\Pi_i)$ for each $i = 0, 1, \ldots, T - 1$.

If we assume that there is a T-round algorithm for Π_0, then by an iterated application of Lemma 9.1, there is a $(T-1)$-round algorithm for Π_1, a $(T-2)$-round algorithm for Π_2, and so on. In particular, there is a 0-round algorithm for Π_T.

Algorithms that run in 0 rounds are much easier to reason about than algorithms in general. Since there is no communication, each active node must simply map its input, essentially its degree, to some output. In particular, we can try to show that there is no 0-round algorithm for Π_T. If this is the case, we have a contradiction with our original assumption: there is no T-round algorithm for Π_0.

We will now proceed to formally define output problems.

9.2.2 Output Problems

For each locally verifiable problem Π we will define a unique *output problem* $\mathrm{re}(\Pi)$.

Let $\Pi_0 = (\Sigma_0, \mathbf{A}_0, \mathbf{P}_0)$ be a bipartite locally verifiable problem on (d, δ)-biregular trees. We define the output problem $\Pi_1 = \mathrm{re}(\Pi_0) = (\Sigma_1, \mathbf{A}_1, \mathbf{P}_1)$ of Π_0 on (δ, d)-biregular trees as follows—note that we swapped the degrees of active vs. passive nodes here.

The alphabet Σ_1 consists of all possible non-empty subsets of Σ_0. The roles of the active and passive nodes are inverted, and new configurations are computed as follows.

(a) The active configurations $\mathbf{A}_1$ consist of all multisets

$$[X_1, X_2, \ldots, X_\delta], \text{ where } X_i \in \Sigma_1 \text{ for all } i = 1, \ldots, \delta,$$

such that for ***every*** choice of $x_1 \in X_1$, $x_2 \in X_2$, …, $x_\delta \in X_\delta$ we have $[x_1, x_2, \ldots, x_\delta] \in \mathbf{P}_0$, i.e., it is a passive configuration of Π_0.

(b) The passive configurations $\mathbf{P}_1$ consist of all multisets

$$[Y_1, Y_2, \ldots, Y_d], \text{ where } Y_i \in \Sigma_1 \text{ for all } i = 1, \ldots, d,$$

for which ***there exists*** a choice $y_1 \in Y_1$, $y_2 \in Y_2$, …, $y_d \in Y_d$ with $[y_1, y_2, \ldots, y_d] \in \mathbf{A}_0$, i.e., it is an active configuration of Π_0.

9.2.3 Example: Weak 3-labeling

To illustrate the definition, let us construct the output problem $\mathrm{re}(\Pi_0) = (\Sigma_1, \mathbf{A}_1, \mathbf{P}_1)$ of weak 3-labeling problem $\Pi_0 = (\Sigma_0, \mathbf{A}_0, \mathbf{P}_0)$. Recall that

$$\begin{aligned}
\Sigma_0 &= \{1, 2, 3\}, \\
\mathbf{A}_0 &= \big\{[1,1,2], [1,1,3], [1,2,2], [1,2,3], [1,3,3], [2,2,3], [2,3,3]\big\}, \\
\mathbf{P}_0 &= \big\{[1,1], [2,2], [3,3]\big\}.
\end{aligned}$$

The alphabet Σ_1 consists of all possible (non-empty) subsets of Σ_0:

$$\Sigma_1 = \big\{\{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\big\}.$$

The active configurations $\mathbf{A}_1$ are all multisets $[X, Y]$ with $X, Y \in \Sigma_1$ such that *all* choices of elements $x \in X$ and $y \in Y$ result in a multiset $[x, y] \in \mathbf{P}_0$. For example $X = \{1\}$ and $Y = \{1, 2\}$ is *not* a valid choice: we could choose $x = 1$ and $y = 2$ to construct $[1, 2] \notin \mathbf{P}_0$. In general, whenever $|X| > 1$ or $|Y| > 1$, we can find $x \in X$, $y \in Y$ with $x \neq y$, and then $[x, y] \notin \mathbf{P}_0$. Therefore the only possibilities are $|X| = |Y| = 1$, and then we must also have $X = Y$. We obtain

$$\mathbf{A}_1 = \big\{\big[\{1\}, \{1\}\big], \big[\{2\}, \{2\}\big], \big[\{3\}, \{3\}\big]\big\}.$$

But since the active configurations only allow singleton sets, we can restrict ourselves to them when listing the possible passive configurations; we obtain simply

$$\mathbf{P}_1 = \Big\{ \big[\{1\},\{1\},\{2\}\big], \big[\{1\},\{1\},\{3\}\big], \big[\{1\},\{2\},\{2\}\big], \big[\{1\},\{2\},\{3\}\big], \\ \big[\{1\},\{3\},\{3\}\big], \big[\{2\},\{2\},\{3\}\big], \big[\{2\},\{3\},\{3\}\big] \Big\}.$$

9.2.4 Complexity of Output Problems

In this section we prove Lemma 9.1 that states that the output problem $\mathrm{re}(\Pi)$ of Π can be solved one round faster than Π.

The proof is by showing that we can truncate the execution of a T-round algorithm and output the set of *possible outputs*. As we will see, this is a solution to the output problem.

Proof of Lemma 9.1. Assume that we can solve some problem $\Pi_0 = (\Sigma_0, \mathbf{A}_0, \mathbf{P}_0)$ on (d,δ)-biregular trees in T rounds using some deterministic PN-algorithm S. We want to design an algorithm that works in (δ,d)-biregular trees and solves $\Pi_1 = \mathrm{re}(\Pi_0)$ in $T-1$ rounds.

Note that we are considering the same family of networks, but we are only switching the sides that are marked as active and passive. We will call these Π_0-*active* and Π_1-*active* sides, respectively.

The algorithm for solving Π_1 works as follows. Let $N = (V, P, p)$ be any port-numbered network with a (δ,d)-biregular tree as the underlying graph. Each Π_1-active node u, in $T-1$ rounds, gathers its full $(T-1)$-neighborhood $\mathrm{ball}_N(u, T-1)$. Now it considers all *possible* outputs of its Π_0-active neighbors under the algorithm S, and outputs these.

Formally, this is done as follows. When N is a port-numbered network, we use $\mathrm{ball}_N(u,r)$ to refer to the information within distance r from node u, including the local inputs of the nodes in this region, as well as the port numbers of the edges connecting nodes within this region. We say that a port-numbered network H is *compatible* with $\mathrm{ball}_N(u,r)$ if there is a node $v \in H$ such that $\mathrm{ball}_H(v,r)$ is isomorphic to $\mathrm{ball}_N(u,r)$.

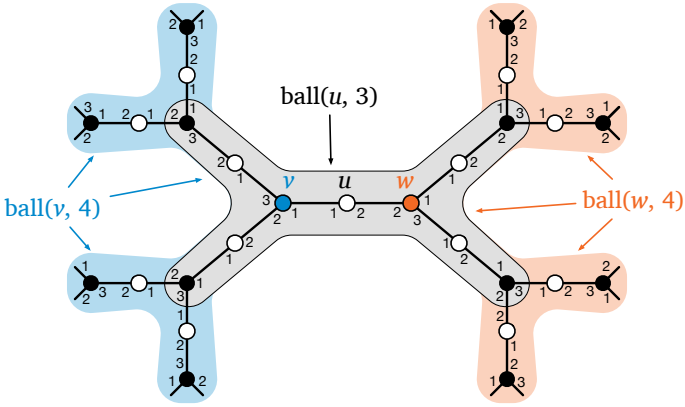

Figure 9.3: Illustration of the round elimination step. A fragment of a $(2,3)$-biregular tree. The 3-neighborhood of node u consists of the gray area. The 4-neighborhoods of nodes v and w consist of the blue and orange areas, respectively. Since the input is a tree, these intersect exactly in the 3-neighborhood of u.

For each neighbor v of u, node u constructs all possible fragments $\mathrm{ball}_H(v,T)$ such that H is compatible with $\mathrm{ball}_N(u,T-1)$ and has a (δ,d)-biregular tree as its underlying graph. Then u simulates the Π_0-algorithm S on $\mathrm{ball}_H(v,T)$. The algorithm outputs some label $x \in \Sigma_0$ on the edge $\{u,v\}$. Node u adds each such label x to set $S(u,v)$; finally node u will label edge $\{u,v\}$ with $S(u,v)$.

By construction, $S(u,v)$ is a nonempty set of labels from Σ_0, i.e., $S(u,v) \in \Sigma_1$. We now prove that the sets $S(u,v)$ form a solution to Π_1. We use the assumption that the underlying graph G is a tree. Let H be any port-numbered network compatible with $\mathrm{ball}_N(u,T-1)$. Consider any two neighbors v and w of u: since there are no cycles, we have

$$\mathrm{ball}_H(v,T) \cap \mathrm{ball}_H(w,T) = \mathrm{ball}_H(u,T-1) = \mathrm{ball}_N(u,T-1).$$

In particular, once $\mathrm{ball}_H(u,T-1)$ is fixed, the outputs of v and w, respectively, depend on the structures of $\mathrm{ball}_H(v,T) \setminus \mathrm{ball}_H(u,T-1)$ and $\mathrm{ball}_H(w,T) \setminus \mathrm{ball}_H(u,T-1)$, which are completely distinct. See Figure 9.3 for an illustration. Therefore, if there exist $x \in S(u,v)$ and

$y \in S(u, w)$, then there exists a port-numbered network H such that running S, node v outputs x on $\{v, u\}$ and node w outputs y on $\{w, u\}$. This further implies that since S is assumed to work correctly on all port-numbered networks, for any combination of $x_1 \in S(u, v_1)$, $x_2 \in S(u, v_2)$, $\ldots$, $x_\delta \in S(u, v_\delta)$, we must have that

$$[x_1, x_2, \ldots, x_\delta] \in \mathbf{P}_0.$$

This implies that

$$[S(u, v_1), S(u, v_2), \ldots, S(u, v_\delta)] \in \mathbf{A}_1.$$

It remains to show that for each Π_0-active node v, it holds that the sets $S(u_1, v)$, $S(u_2, v)$, $\ldots$, $S(u_d, v)$, where u_i are neighbors of v, form a configuration in $\mathbf{P}_1$. To see this, note that the Π_1-active nodes u_i simulate S on every port-numbered fragment, including the true neighborhood $\text{ball}_N(v, T)$ of v. This implies that the output of v on $\{v, u_i\}$ running S in network N is included in $S(u_i, v)$. Since S is assumed to be a correct algorithm, these true outputs $x_1 \in S(u_1, v)$, $x_2 \in S(u_2, v)$, $\ldots$, $x_d \in S(u_d, v)$ form a configuration

$$[x_1, x_2, \ldots, x_d] \in \mathbf{A}_0,$$

which implies that

$$[S(u_1, v), S(u_2, v), \ldots, S(u_d, v)] \in \mathbf{P}_1,$$

as required. □

9.2.5 Example: Complexity of Weak 3-labeling

Now we will apply the round elimination technique to show that the weak 3-labeling problem is not solvable in 1 round. To do this, we show that the output problem of weak 3-labeling is not solvable in 0 rounds.

Lemma 9.2. *Weak 3-labeling is not solvable in 1 round in the* PN*-model on* $(3, 2)$*-biregular trees.*

Proof. In Section 9.2.3 we saw the output problem of weak 3-labeling. We will now show that this problem is not solvable in 0 rounds on $(2,3)$-biregular trees. By Lemma 9.1, weak 3-labeling is then not solvable in 1 round on $(3,2)$-biregular trees. Let $\Pi_1 = (\Sigma_1, \mathbf{A}_1, \mathbf{P}_1)$ denote the output problem of weak 3-labeling.

In a 0-round algorithm an active node v sees only its own side (active or passive) and its own port numbers. Since

$$\mathbf{A}_1 = \big\{ [\{1\},\{1\}], [\{2\},\{2\}], [\{3\},\{3\}] \big\},$$

each active node v must output the same label $X \in \big\{\{1\},\{2\},\{3\}\big\}$ on both of its incident edges.

Since all active nodes look the same, they all label their incident edges with exactly one label X. Since $[X,X,X]$ is not in $\mathbf{P}_1$ for any $X \in \Sigma_1$, we have proven the claim. □

9.2.6 Example: Iterated Round Elimination

We will finish this chapter by applying round elimination *twice* to weak 3-labeling. We will see that the problem

$$\Pi_2 = \mathrm{re}(\Pi_1) = \mathrm{re}(\mathrm{re}(\Pi_0))$$

obtained this way *is* 0-round solvable.

Let us first construct Π_2. Note that this is again a problem on $(3,2)$-biregular trees. We first *simplify* notation slightly; the labels of Π_1 are sets and labels of Π_2 would be sets of sets, which gets awkward to write down. But the configurations in Π_1 only used *singleton* sets. Therefore we can *leave out* all non-singleton sets without changing the problem, and then we can *rename* each singleton set $\{x\}$ to x. After these simplifications, we have got

$$\Sigma_1 = \{1,2,3\},$$
$$\mathbf{A}_1 = \big\{ [1,1], [2,2], [3,3] \big\},$$
$$\mathbf{P}_1 = \big\{ [1,1,2], [1,1,3], [1,2,2], [1,2,3], [1,3,3], [2,2,3], [2,3,3] \big\}.$$

Alphabet Σ_2 consists of all non-empty subsets of Σ_1, that is

$$\Sigma_2 = \big\{\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\big\}.$$

The active configurations are all multisets $[X_1, X_2, X_3]$ where $X_1, X_2, X_3 \in \Sigma_2$ such that *any* way of choosing $x_1 \in X_1$, $x_2 \in X_2$, and $x_3 \in X_3$ is a configuration in $\mathbf{P}_1$. There are many cases to check, the following observation will help here (the proof is left as Exercise 9.4):

- $\mathbf{P}_1$ consists of all 3-element multisets over Σ_1 where at least one of the elements is not 1, at least one of the elements is not 2, and at least one of the elements is not 3.
- It follows that $\mathbf{A}_2$ consists of all 3-element multisets over Σ_2 where at least one of the elements does not contain 1, at least one of the elements does not contain 2, and at least one of the elements does not contain 3.

It follows that we can enumerate all possible configurations e.g. as follows (here $X, Y, Z \in \Sigma_2$):

$$\begin{aligned}\mathbf{A}_2 = &\Big\{\big[X,Y,Z\big] \,\Big|\, X \subseteq \{1,2\}, Y \subseteq \{1,3\}, Z \subseteq \{2,3\}\Big\} \\ \cup &\Big\{\big[X,Y,Z\big] \,\Big|\, X \subseteq \{1\}, Y \subseteq \{2,3\}, Z \subseteq \{1,2,3\}\Big\} \\ \cup &\Big\{\big[X,Y,Z\big] \,\Big|\, X \subseteq \{2\}, Y \subseteq \{1,3\}, Z \subseteq \{1,2,3\}\Big\} \\ \cup &\Big\{\big[X,Y,Z\big] \,\Big|\, X \subseteq \{3\}, Y \subseteq \{1,2\}, Z \subseteq \{1,2,3\}\Big\}.\end{aligned} \tag{9.1}$$

On the passive side, $\mathbf{P}_2$ consists of all multisets $[X, Y]$ where we can choose $x \in X$ and $y \in Y$ with $[x, y] \in \mathbf{A}_1$. But $[x, y] \in \mathbf{A}_1$ is equivalent to $x = y$, and hence $\mathbf{P}_2$ consists of all multisets $[X, Y]$ where we can choose some $x \in X$ and choose the same value $x \in Y$. Put otherwise,

$$\mathbf{P}_2 = \Big\{\big[X,Y\big] \,\Big|\, X \in \Sigma_2, Y \in \Sigma_2, X \cap Y \neq \varnothing\Big\}.$$

Lemma 9.3. *Let Π_0 denote the weak 3-labeling problem. The problem $\Pi_2 = \mathrm{re}(\mathrm{re}(\Pi_0)) = (\Sigma_2, \mathbf{A}_2, \mathbf{P}_2)$ is solvable in 0 rounds.*

Proof. The active nodes always choose the configuration

$$\big[\{1,2\},\{1,3\},\{2,3\}\big] \in \mathbf{A}_2$$

and assign the sets in some way using the port numbers, e.g., the edge incident to port 1 is labeled with $\{2,3\}$, the edge incident to port 2 is labeled with $\{1,3\}$, and the edge incident to port 3 is labeled with $\{1,2\}$.

Since each pair of these sets has a non-empty intersection, no matter which sets are assigned to the incident edges of passive nodes, these form a valid passive configuration in $\mathbf{P}_2$. □

9.3 Quiz

Consider the following bipartite locally verifiable labeling problem $\Pi = (\Sigma, \mathbf{A}, \mathbf{P})$ on $(2,2)$-biregular trees:

$$\begin{aligned}
\Sigma &= \{1,2,3,4,5,6\},\\
\mathbf{A} &= \big\{[1,6],[2,5],[3,4]\big\}, \text{ and}\\
\mathbf{P} &= \big\{[x,y] \,\big|\, x \in \{3,5,6\}, y \in \{1,2,3,4,5,6\}\big\}\\
&\quad \cup \big\{[x,y] \,\big|\, x \in \{4,5,6\}, y \in \{2,3,4,5,6\}\big\}.
\end{aligned}$$

Give a 0-round algorithm for solving Π.

9.4 Exercises

Exercise 9.1 (encoding graph problems)**.** Even if a graph problem is defined for general (not bipartite) graphs, we can often represent it in the bipartite formalism. If we take a d-regular tree G and subdivide each edge, we arrive at a $(d,2)$-biregular tree H, where the active nodes represent the nodes of G and passive nodes represent the edges of G.

Use this idea to encode the following graph problems as bipartite locally verifiable labelings in $(d,2)$-biregular trees. Give a brief explanation of why your encoding is equivalent to the original problem. You

can ignore the leaf nodes and their constraints; it is sufficient to specify constraints for the active nodes of degree d and passive nodes of degree 2.

(a) Vertex coloring with $d+1$ colors.
(b) Maximal matching.
(c) Dominating set.
(d) Minimal dominating set.

Exercise 9.2 (algorithms in the bipartite model)**.** The bipartite model can be used to run algorithms from the standard PN and LOCAL models. Using the idea of Exercise 9.1, we encode the *maximal independent set* problem in 3-regular trees as the following bipartite locally verifiable problem $\Pi = (\Sigma, \mathbf{A}, \mathbf{P})$ in $(3,2)$-biregular trees:

$$\Sigma = \{\mathsf{I}, \mathsf{O}, \mathsf{P}\},$$
$$\mathbf{A} = \big\{[\mathsf{I}, \mathsf{I}, \mathsf{I}], [\mathsf{P}, \mathsf{O}, \mathsf{O}]\big\},$$
$$\mathbf{P} = \big\{[\mathsf{I}, \mathsf{P}], [\mathsf{I}, \mathsf{O}], [\mathsf{O}, \mathsf{O}]\big\}.$$

In $\mathbf{A}$, the first configuration corresponds to a node in the independent set X, and the second configuration to a node not in X. A node not in X points to a neighboring active node with the label P: the node pointed to has to be in X. The passive configurations ensure that two active nodes connected by a passive node are not both in X, and that the pointer P always points to a node in X.

Assume that the active nodes are given a 4-coloring c as input. That is, $c\colon V_A \to \{1,2,3,4\}$ satisfies $c(v) \neq c(u)$ whenever the active nodes $v, u \in V_A$ share a passive neighbor $w \in V_P$. The nodes also know whether they are active or passive, but the nodes do not have any other information.

Present a PN-algorithm in the state machine formalism for solving Π. Prove that your algorithm is correct. What is its running time? How does it compare to the complexity of solving maximal independent set in the PN model, given a 4-coloring?

Exercise 9.3 (Round Eliminator)**.** There is a computer program, called Round Eliminator, that implements the round elimination technique and that you can try out in a web browser:

https://github.com/olidennis/round-eliminator

Let Π_0 be the weak 3-labeling problem defined in Section 9.1.2. Use the Round Eliminator to find out what are $\Pi_1 = \mathrm{re}(\Pi_0)$ and $\Pi_2 = \mathrm{re}(\Pi_1)$. In your answer you need to show how to encode Π_0 in a format that is suitable for the Round Eliminator, what were the answers you got from the Round Eliminator, and how to turn the answers back into our usual mathematical formalism.

Exercise 9.4 (iterated round elimination)**.** Fill in the missing details in Section 9.2.6 to show that formula (9.1) is a correct definition of the active configurations for problem Π_2 (i.e., it contains all possible configurations and only them).

Exercise 9.5 (solving weak 3-labeling)**.** Present a 2-round deterministic PN-algorithm for solving weak 3-labeling in $(3, 2)$-biregular trees.

Exercise 9.6 (sinkless orientation)**.** Consider the sinkless orientation problem, denoted by Π, on $(3, 3)$-biregular trees from Section 9.1.2. Compute the output problems $\mathrm{re}(\Pi)$ and $\mathrm{re}(\mathrm{re}(\Pi))$; include a justification for your results.

9.5 Bibliographic Notes

Linial's [3] lower bound for vertex coloring in cycles already used a proof technique that is similar to round elimination. However, for a long time it was thought that this is an ad-hoc technique that is only applicable to this specific problem. This started to change in 2016, when the same idea found another very different application [2]. Round elimination as a general-purpose technique was defined and formalized by Brandt [1] in 2019, and implemented as a computer program by Olivetti [4].

9.6 Bibliography

[1] Sebastian Brandt. An automatic speedup theorem for distributed problems. In *Proc. 38th ACM Symposium on Principles of Dis-*

tributed Computing (PODC 2019), 2019. arXiv:1902.09958, doi:10.1145/3293611.3331611.

[2] Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela, and Jara Uitto. A lower bound for the distributed Lovász local lemma. In *Proc. 48th ACM Symposium on Theory of Computing (STOC 2016)*, 2016. arXiv:1511.00900, doi:10.1145/2897518.2897570.

[3] Nathan Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992. doi:10.1137/0221015.

[4] Dennis Olivetti. Round Eliminator: a tool for automatic speedup simulation, 2020. URL: https://github.com/olidennis/round-eliminator.